Real-time High-Resolution Video Object Segmentation

Wang Duolin
Software Engineering, School of Computer and Software
Nanjing University of Information Science and Technology
darren_wangl@foxmail.com

Chen Yadang
Engineering Research Center of Digital Forensics, Ministry of Education, School of Computer and Software
Nanjing University of Information Science and Technology
adamchen@nuist.edu.cn

Yang Zhi-Xin
State Key Laboratory of Internet of Things for Smart City, Department of Electromechanical Engineering
University of Macau
zxyang@um.edu.mo

Wu Enhua
State Key Laboratory of Computer Science, Institute of Software, University of Chinese Academy of Sciences
University of Macau
ehwu@um.edu.mo

Abstract

Semi-supervised video object segmentation aims to segment a specific object throughout a video sequence with only the first-frame annotation. Though recent state-of-the-art methods have achieved high accuracy, the expensive computation cost and heavy memory usage are impractical for most applications. In this paper, we propose an efficient semi-supervised video object segmentation approach that processes the high-resolution video cases at real-time speed with less memory usage. Specifically, we encode video frames and masks independently in the encoding stage, significantly reducing the computation cost caused by large, repeated encoding in previous state-of-the-art methods. On the other hand, we use a lightweight global memory module to represent the feature extracted from all the past video frames, which can be easily updated automatically over time. In addition, to reduce the confusion caused by distractors, the prediction masks of the previous frames are used to constrain the segmentation location of the foreground, and a channel-wise attention operation is adopted to enhance the representation of all foreground objects. In the experiments, our method achieves state-of-the-art on both the DAVIS and YouTube-VOS datasets. It has an accuracy of 84.9% and an average single-frame segmentation speed of $0.04s$ on the DAVIS 2017 dataset. The results demonstrate the proposed method significantly reduces the computation cost and memory usage for high-resolution video cases while maintaining a competitive accuracy compared with other recent methods.

***Keywords:** Video Object Segmentation, Semantic Segmentation, Channel-wise Attention, Spatial Constraint.*

1. Introduction

Semi-supervised Video Object Segmentation (semi-VOS) separates all foreground objects from the background of the entire video sequence using the ground truth of the first frame and has provided reliable and efficient segmentation algorithms for video classification [7, 30], detection [6, 12, 19], and reconstruction [20, 32, 46]. To increase the commercial value, VOS must be enhanced to support high-resolution videos and provide high-quality segmentation results efficiently.

With the advancement of deep learning, semi-VOS have made significant progress. To make the best possible use of the information provided by ground truth, the initial-based methods [2, 3, 9, 22, 35] use the first frame of ground truth to complete segmentation quickly. However, the segmentation performance degrades drastically when they encounter violent object deformation. An alternative approach

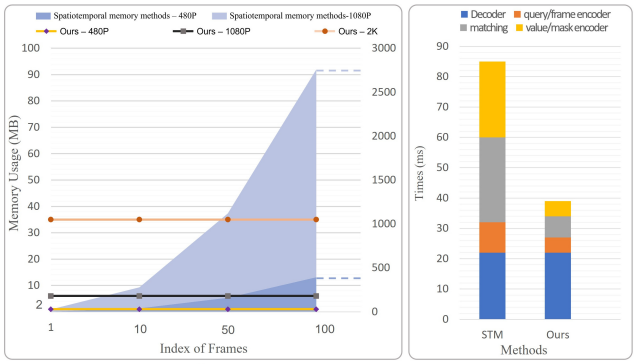


Figure 1. (a) Comparison of memory consumption between our method and the spatiotemporal memory methods at different resolutions. The memory consumption of the spatiotemporal memory methods increases exponentially with the growth of the video memory (i.e., the past segmented frames) and image resolution.However, our method keeps a low and fixed memory usage at various resolutions. (b) Comparison of the computation time of our method with STM [25] for single-frame segmentation. Our model is 50% faster than STM [25] when segmenting a single frame.

is to use the previous segmentation result to guide the prediction of the current frame, known as mask propagation [11,13,26,38], which is susceptible to interference from occlusion. The recently proposed segmentation methods based on spatiotemporal memory networks [4, 14, 25, 37] provide high-quality results. However, they cannot operate in real-time at high-resolution and are limited by the heavy computational cost with the growth of video memory (i.e. the number of past segmented frames). Fig. 1(a) compares the memory consumption of our method and the spatiotemporal memory methods at different resolutions. The memory consumption of the spatiotemporal memory methods increases exponentially of the growth of the video memory (i.e., the past segmented frames) and image resolution. However, our method keeps a low and fixed memory usage at various resolutions. Fig. 1(b) compares the computing time of our method with STM [25] for single-frame segmentation. Our model is 50% faster than STM [25] when segmenting a single frame.

Designing a real-time VOS algorithm at high resolution is very challenging, and so far, excellent VOS algorithms are limited to training and testing at lower resolutions $(480P)$. Neural network training at high resolution becomes slow and consumes many computational resources. We propose an efficient segmentation model called Global Attention Module (GAM) based on the spatiotemporal memory algorithm [14, 25, 29] to solve the above problem. Fig. 2 shows how we can reduce resource consumption. The spatiotemporal memory approach combines and encodes video frames with their masks to generate key-value pairs and stores them in a memory pool in a stacked

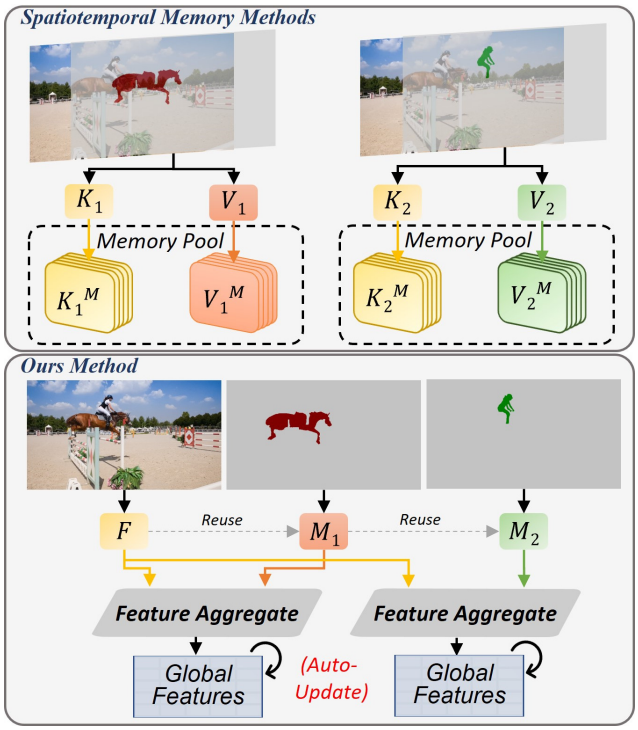


Figure 2. Comparison between our method and the spatiotemporal memory method in the feature extraction and storage stages. Unlike storing the key-value features extracted from frames and masks into the memory pool, we encode frames and masks separately and aggregate the extracted information into the global feature.

manner, which is limited by the size of memory pool. Consequently, the model periodically discards the information in the memory pool, which means that not all memory nodes participate in matching the current frame. In contrast, we input frames and masks into two independent encoders to obtain the frame feature F and the mask feature M. Then we establish a correspondence between the frame and mask features and aggregate them into a fixed-size global feature, which updates automatically over time. The update of the global attention module will let every memory node participate in matching the current frame. Thus our method ensures that the global feature module learns enough valid information while reducing the memory usage and the computation during feature matching.

Thanks to the above operations, as shown in Fig. 1, we can maintain low memory usage in high-resolution videos and achieve segmentation speeds of 25 FPS and excellent segmentation results on $2K$ videos with only a modern GPU compared to the spatiotemporal memory algorithm. To further improve the segmentation performance of the model and mitigate the mis-segmentation problem caused by distractors, we introduce a Channel-wise Attention Module (CAM, Fig. 5) in the matching process, which enhances

the representation of all foreground objects. We also use the Mask Embedding Module (MEM, Fig. 6), which spatially constrains the segmentation range of the current frame using the segmentation masks of the previous frames and their position encoding, which significantly reduces the distracting problem.

In summary, our contributions are as follows.

1. Our model reduces memory consumption by 50% compared to temporal memory algorithms when segmenting a single frame.

2. Our method also maintains real-time computation when dealing with high-resolution and long video sequences.

3. Using the channel attention module and mask embedding module, the former enhances the representation of foreground objects, and the latter spatially constrains the segmentation range of foreground objects.

2. Related Works

2.1. Traditional Methods

The initial-based methods [2, 3, 9, 22, 35] rely on the ground truth provided by the first frame for fine-tuning, which requires establishing a powerful pixel-level deep learning model to achieve fast and efficient segmentation of frames. OSVOS [2] is representative of this type of algorithm. He processes each frame independently and does not impose constraints on timing or space. Its advantage is that the segmentation speed is swift. Unfortunately, its segmentation accuracy is not optimal and is susceptible to the interference of similar objects. OnAVOS [35] and OSVOS-S [22] introduce the concept of online learning based on OSVOS [2]. PML [3] processes each frame through a single forward path and nearest neighbor search embedded in the network. VidMatch [9] proposed the concept of a similarity map, which provides strong interpretability for target segmentation and ideas for subsequent algorithms. Many initial-based methods require online training, so the model consumes many resources in fine-tuning, which seriously drags down the calculation speed and makes it difficult for the algorithm to balance calculation time and accuracy.

When segmenting the current frame based on the propagation methods [11, 13, 26, 38], the mask that has completed the pixel-level annotation of the previous frame is introduced, and the output of the previous frame is directed to the target in the current frame. MaskTrack [26] is a representative algorithm that considers the timing information of the video. MaskTrack [26] uses the rough mask generated in the previous frame and sends it to the training network to obtain the estimate of the current frame. Since there will be no large displacement between two adjacent frames, such rough results will produce a reasonable estimate. Lucid [11] introduces the concept of data augmentation, which increases the training data by rotating and cropping the image to avoid model overfitting. Joint-task [13] and learning-correspondence [38] use KNN [47] to learn correspondence from periodic consistency of time.

2.2. Spatiotemporal Memory Methods

The spatiotemporal memory methods [4, 14, 18, 25, 29, 37, 41, 48] consist of two steps. First, in the time dimension, the algorithm saves the features extracted from all memory nodes in the memory pool. Second, in the spatial dimension, the algorithm uses the rich features in the memory pool and the local features extracted from the current frame to perform pixel-level matching to determine whether each pixel belongs to the foreground or background element. STM [25] is a pioneering algorithm in this field. Although STM [25] has excellent segmentation performance, it suffers from a huge computational resource overhead. If it saves the information from all memory nodes, the memory pool will increase, eventually leading to memory overflow. Therefore, STM [25] will periodically discard the previously learned feature information, contrary to the algorithm's original intention.

There are many improved algorithms based on spatiotemporal memory methods [4, 14, 29, 37, 41, 45, 48]. In order to improve the segmentation speed, GC [14] transforms the memory pool into a fixed-size global matrix, which solves the problem of memory overflow and improves the segmentation speed. The core of Swiftnet [37] is to reduce the spatial and temporal redundancy of video segmentation through pixel-adaptive memory. In the temporal dimension, pixel-adaptive memory triggers a memory update when the target changes between frames. Spatially, pixel-adaptive memory selectively performs memory updates and matchings only on dynamic pixels, ignoring static pixels. This feature significantly reduces redundant computation on segmentation-independent pixels. STCN [4] can directly obtain the correspondence between frames without repeatedly encoding the mask features of each target object. To alleviate the interference of similar objects, RMNet [41] uses the optical flow [31, 44] to constrain the segmentation range of the current frame to reduce the interference of distractors. KMN [29] performs not only local matching from the memory pool to the current frame but also performs global matching from the current frame to the memory pool. This strategy dramatically alleviates the interference of similar objects. SCM [48] uses the segmentation mask of the previous frame to constrain the segmentation range of the current frame spatially. LCM [8] and BMVOS [5] use the prediction mask of the previous frame and the ground truth of the first frame to learn the object position information and reduce the accumulated error in the segmentation process.

In order to improve the slow operation speed of the spatiotemporal memory methods [5, 8, 25, 29, 48, 48], in the

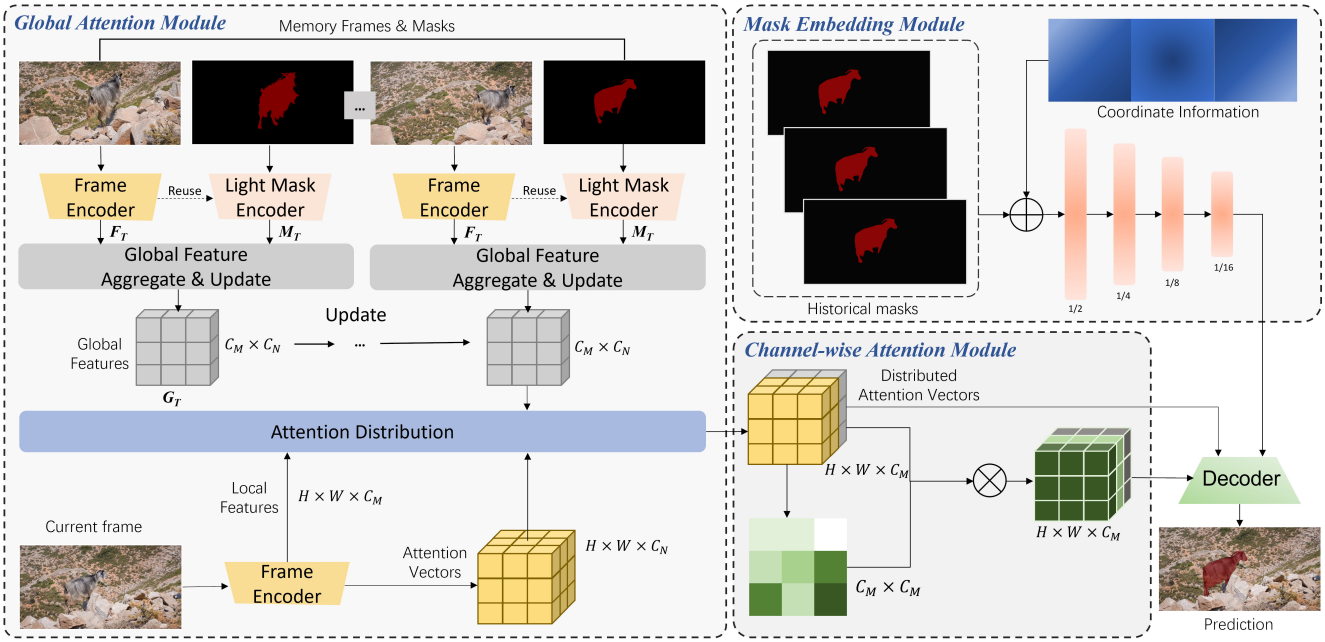


Figure 3. Pipeline containing three modules. Past frames and masks are input into the network for encoding and aggregated into global features in the global attention module, which updates automatically over time. The encoder also generates local features from the current frame. We match the global features with the local features to get the distributed attention vectors, then pass the attention vectors to the channel attention module, whose result feeds to the decoder along with the result of the mask embedding module to produce the segmentation result for the frame.

encoding stage, we independently encode the video frame and the mask, which significantly reduces the resource consumption caused by the repeated calculation of mask features in multi-target scenarios. At the same time, we use a lightweight fixed-size global feature module to save the feature information extracted from all memory nodes, which is updates automatically over time. Due to the high redundancy of videos, our model maintains the real-time segmentation speed with little loss in the accuracy of the results. To optimize the segmentation performance and reduce the interference of similar objects, we feed the matching results of the current frame into the channel attention module to enhance the representation of all foreground objects. At the same time, the prediction masks of the previous frames constrain the spatial segmentation range of foreground objects.

To sum up, the essence of the spatiotemporal memory algorithm [4, 14, 25, 29, 37, 41, 48] is to use the information learned from all past nodes to match the current frame. Our approach inherits this advantage and dramatically improves the segmentation speed, which can segment high-resolution videos in real time.

3. Methods

3.1. Overview

Fig. 3 shows our general framework, which consists of three modules. The first one is the Global Attention Module (GAM). For the memory nodes (including the ground truth of the first frame), we input frames and masks into two independent encoders to get the frame features F_T and mask features M_T. Then we establish the correspondence between F_T and M_T and aggregate them to get global features, which update automatically over time. We input the current frame into the frame encoder to get the local features. Then the global features and local features are subjected to the attention matching operation to obtain the attention distribution features, which will be input to the Channel-wise Attention Module (CAM). The result of this process is input to the decoder. At the same time, we input

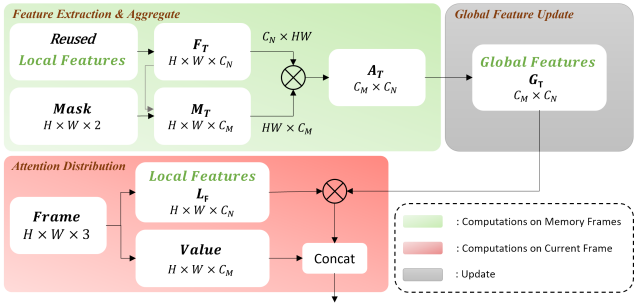


Figure 4. Attention distribution. We match the local features extracted from the current frame with the global features extracted from the past nodes and output them as distributed attention vectors.

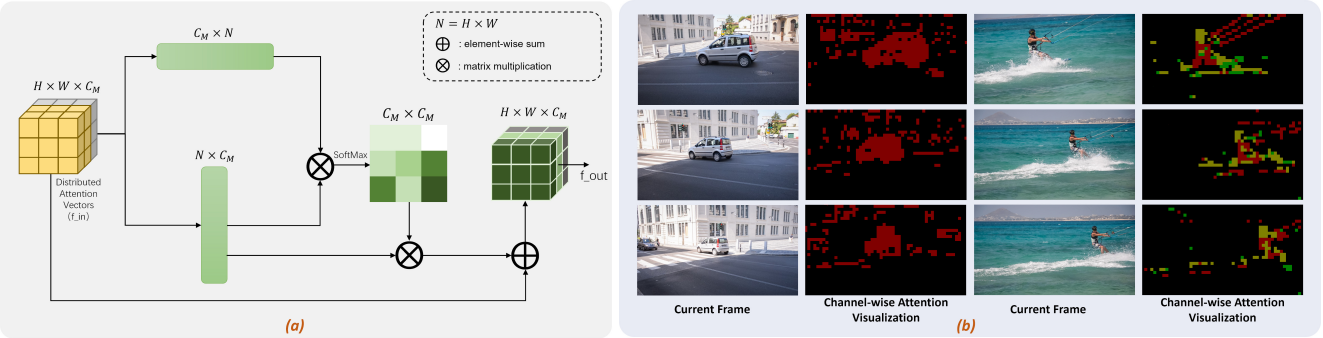


Figure 5. (a) Schematic diagram of the channel-wise attention module. Distributed attention vectors as input, f_{in}. The channel-wise attention map is calculated from f_{in}, representing the channel's inner relationship. Finally, the channel attention map is matched with f_{in} to reassign the weights of each channel.(b) Resulting diagram of the channel-wise attention module.

the history mask of previous frames and its position encoding into the Mask Embedding Module (MEM), and put the result of this module into the decoder. Finally, the decoder calculates the segmentation prediction of the current frame.

3.2. Global Attention Module

3.2.1 Spatiotemporal Memory Methods Versus Ours

The spatiotemporal memory approach brings a series of SOTA solutions for VOS. In the memory node, as Fig. 4 shows, it inputs the frame and mask into the memory encoder to get $F_T \in R^{HW \times C}$, $M_T \in R^{HW \times C}$ It inputs the current frame into the query encoder to get $Q_R \in R^{HW \times C}$, where H, W, and C represent the length, width, and the number of channels of the input image, respectively. The K_y and V_l extracted from the memory node are combined by concatenating as the number of frames advances to form $K_Y^M \in R^{THW \times C}$, and $V_l^M \in R^{THW \times C}$, where T represents the number of frames. The memory consumption and computation speed of the model are sensitive to the change of T. At this point, the spatiotemporal memory methods will have to discard some memory nodes to ensure that the memory will not overflow, so not all memory nodes participate in the current frame segmentation, which is inconsistent with the intent of the algorithm.

In contrast, our approach encodes frames and masks separately and inputs frame features F_T and mask features M_T into a global module of fixed size. Our updating algorithm allows all memory nodes to update the current frame. Thus, our algorithm has a representational power similar to the spatiotemporal memory approach while maintaining efficient computation.

3.2.2 Feature Extraction and Reuse

For the memory node, we input the frame into the ResNet-50-based frame encoder. As Fig. 4 shows, and we define the frame feature as $F_T \in R^{HW \times C}$. Then we input F_T and mask into the ResNet-18-based mask encoder, we define the mask feature as $M_T \in R^{HW \times C}$.The model only needs to encode the video frames once. This scheme significantly reduces the resource consumption caused by repeated computation of mask features. At the same time, we experimentally demonstrate that the masks do not require deep neural networks to extract enough valid information, so we use the lightweight ResNet-18 as the mask encoder. These initiatives significantly reduce the computational requirements of the model.

For the current frame, we define the output of the frame encoder as $L_F \in R^{H \times W \times C_N}$. The output of value encoder is $V_L \in R^{H \times W \times C_M}$. Since all video frames use the same encoder, each frame only needs to be encoded once during the propagation process. After the query operation, the "query frame feature" can be reused as the "memory frame feature" to reduce the computational redundancy.

3.2.3 Global Features Aggregation and Update

GAM establishes a relationship between frame features F_T and mask features M_T. Then aggregating F_T and M_T into a fixed-size global feature G_T, which can be regarded as $C_M(C_N)$ single-channel feature maps, and then treat them as weight matrices associated with a vector of key-value pairs. The aggregation formula is

$$A_T = (F_E(I))^T M_E(F_T, M), \tag{1}$$

where A_T denotes the global features at time T, F_E and M_E are frame encoder and mask encoder respectively, I and M are input frame and mask respectively, and F_T is the frame features of this frame. We then include the resulting information in F_T in the global context matrix. Since the global features have fixed sizes, we do so without additional resources. The update for the global context module is as follows:

$$G_T = \frac{1}{T} A_T + \frac{T-1}{T} G_{T-1}, \tag{2}$$

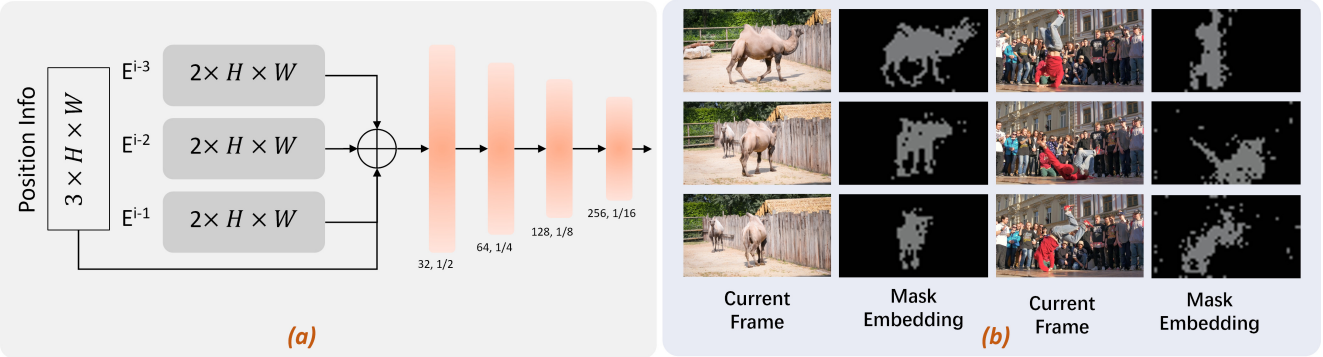


Figure 6. (a) Schematic diagram of the mask embedding module. By providing multiple historic masks simultaneously, the model obtains the position-based coarse prediction for the current frame's target location. (b) Resulting diagram of the mask embedding module.

where G_T denotes the global attention feature. The weights ensure each A_p for $1 \leq p \leq T$ contributes equally to G_T.

3.2.4 Attention Distribution

We match the local information extracted from the current frame to the global information, which we call context distribution. In this process, we multiply the local features with size $H \times W \times C_N$ by global features which have size $C_M \times C_N$ to get a matrix of size $H \times W \times C_M$, and then concatenate the matrix with the value produced by the current frame to get the output of GAM. This may be written:

$$D_T = L_F G_{T-1}, \tag{3}$$

where D_T represents the distributed global features for frame T. We regard the local feature L_F as a set of weighted weights for the global feature G_{T-1}. Since the global feature G_{T-1} has aggregated the area of interest for L_F of the current frame, the weight of the global feature is calculated by Eqn. 3 as the distributed context information corresponding to the current frame. Finally, we sum D_T and the attention vectors of the current frame at the pixel level to get the final output of GAM. GAM summarizes the areas of semantic interest in the query position of the current frame for context features in past frames.

Interestingly, our method can maintain excellent segmentation performance while notably reducing computational resources and shortening segmentation time. The global features store the global summaries of all historical nodes with similar semantic information. i.e., when a pixel is interested in a foreground object, the weight of the foreground object will be continuously strengthened in the historical nodes. In this way, when we add the weight of the local feature to the global feature, we can confirm the target to be queried. Benefiting from the above operations, our method and the spatiotemporal memory methods have similar representational power.

3.3. Channel-wise Attention Module

The output of GAM $f_{out} \in \mathbb{R}^{H \times W \times C_M}$ can complete the segmentation of the current frame admirably. Due to the high redundancy of video information, f_{out} contains much irrelevant information. Directly inputting f_{out} into the decoder may cause the irrelevant information to be amplified and the valuable information to be suppressed. Therefore, to further strengthen the representation of foreground pixels and obtain more valuable and helpful feature information, we adopt a channel-wise attention module. The module details appear in Fig. 5(a), and the input of the module is the output of the GAM, $f_{in} \in \mathbb{R}^{H \times W \times C_M}$. To calculate the interdependencies between channels, f_{in} is reshaped into two feature matrices, $f_1 \in C_M \times N$ and $f_2 \in N \times C_M$, where N means $H \times W$. Then we calculate the channel-wise feature map of f_{in} by:

$$R = f_1 f_2, \tag{4}$$

$$R'(i,j) = \frac{\exp(R(i,j))}{\sum_{i=1}^{N} \exp(R(i,j))} \tag{5}$$

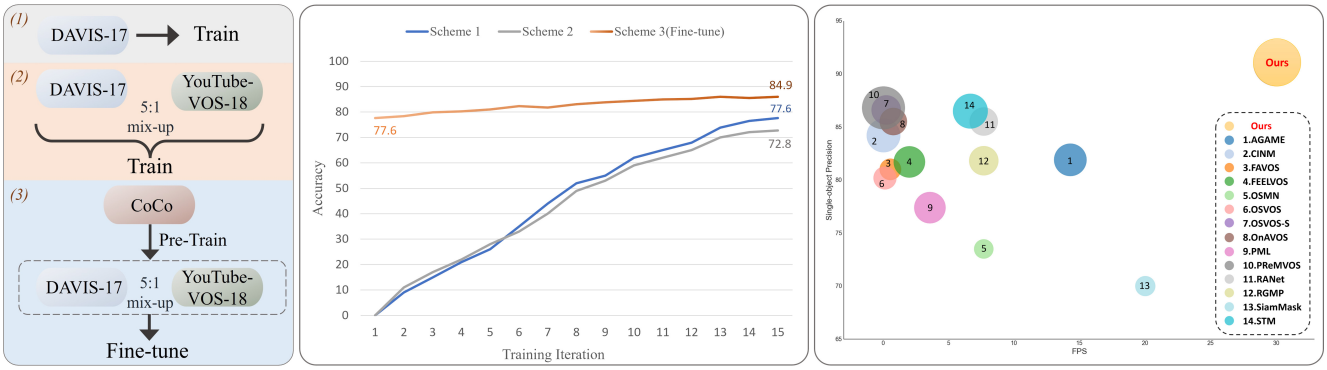
where $R(i,j)$ denotes the relationship between ith channel and jth channel of f_{in}. Then we matrix multiply the channel-wise attention map with the reconstructed features f_2. This process redistributes the attention weights between the channels of fin, which is expressed by the formula:

$$f_R = f_{in} R' \tag{6}$$

Where $f_R \in \mathbb{R}^{N \times C_M}$ represents the input features after redistribution of attention,then we reshape f_R to the size of $\mathbb{R}^{H \times W \times C_M}$, and sum it to f_{in} at the pixel level. Finally, the output of CAM is obtained, which is expressed as:

$$f_{out} = \alpha f_R + f_{in} \tag{7}$$

where $\alpha \geq 0$ is a learned parameter. Visualizations of the output feature map of the CAM appear in Fig. 5(b). We can see that all foreground pixels are enhanced. As shown in



(a) (b) (c)

Figure 7. (a) Comparison of three training strategies. (b) Accuracy variation curves of our method under three different training strategies. (c) Comparison of computing time, single-object accuracy, and multi-object accuracy of our method and other SOTA VOS algorithms. The horizontal coordinate represents frames per second. The vertical coordinate represents each model's accuracy under the single-object dataset. The bubble size represents the accuracy of each model under the multi-model dataset. The more significant the bubble indicates greater progress.

columns 1 and 2, the pixels of the car are enhanced in our CAM visualization. The surfing graph shown in columns 3 and 4 has enhanced representations of the surfer, the surfboard, and the rope in hand.

3.4. Mask Embedding Module

Although GAM has achieved an excellent segmentation effect, since GAM belongs to global matching and is wholly based on visual information, it cannot cope well with the interference of distractors. When there are multiple similar objects in the same frame, GAM cannot accurately distinguish who is the main object. At this point, we need an excellent constraint module to deal with the above problems. Generally, since the spatial position of the target object is close between adjacent frames, the constraint module uses the segmentation mask of the previous frame as a spatial constraint of the current frame. In order to enhance the robustness of this module and better cope with the violent deformation or fast motion of objects in a short time, we adopt a mask embedding module, and MEM takes the prediction masks of the previous frames and their corresponding position information as input, which is expressed by the formula:

$$E^{i-3} \oplus E^{i-2} \oplus E^{i-1} \oplus P \tag{8}$$

where E^i denotes estimated mask of previous frame, P is position information and $\oplus$ indicates concatenation along channel dimension. The details of the mask embedding module appear in Fig. 6(a), we utilize the segmentation masks E^i of previous frames as a spatial constraint for the current segmentation, $E^i \in \mathbb{R}^{2\times H\times W}$. This operation integrates an coarse spatial position of the target object in the previous frames. At the same time, we also use three different scales of position information P, $P \in \mathbb{R}^{3\times H\times W}$, to provide more convincing results, where the three dimensions represent the width, height, and the distance from the center point, respectively. Using the coordinate information can help the model understand the object's spatial position. We concatenate E^i and P into four convolutional layers with ReLU activation functions. Finally, we obtain the features with the spatial information of the target, and we then input the embedded features into the decoder. For the decoder, we use the same structure as STM [25]. Fig. 6(b) shows the intermediate result of MEM. The module limits the segmentation area to the vicinity of the motion range of the target object in the previous few frames, which well avoids the interference of distractors.

4. Experiments

We used DAVIS 2016 as a single-target dataset and DAVIS 2017 and YouTube-VOS as a multi-target datasets. We scaled up the DAVIS 2017 dataset from $480p$ resolution to $1080p$ and $2k$ resolution to test the model's effectiveness when dealing with complex video types (multi-target, high resolution, high frame rate). We refer to these two scaled-up datasets as DAVIS-$1080p$ and DAVIS-$2K$. Evaluation metrics used for object segmentation are average region similarity ($\mathcal{J}$ mean), average contour accuracy ($\mathcal{F}$ mean), and the average of the two ($\mathcal{J}\&\mathcal{F}$ mean). Fig. 7(c) shows that the horizontal coordinate represents frames per second. The vertical coordinate represents each model's accuracy under the single-object dataset. The bubble size represents the accuracy of each model under the multi-model dataset. The more significant the bubble indicates greater progress. Our method is at SOTA level in different segmentation scenarios while maintaining real-time speed.

Figure 8. Results with those from STM [25], BMVOS [5], JOINT [23] and RANet [39]. It can be seen that compared with other methods, our method is very accurate for the boundary segmentation of the target and is not easily affected by distractors.

Table 1. Accuracy of Models with and without Channel Attention Module (CAM).

Variant	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	Time (s)
CAM	82.4	87.4	84.9	0.04
W/O CAM	80.4	85.3	83.3	0.04

Table 2. Accuracy of Models with and without Mask Embedding Module(MEM).

Variant	$\mathcal{J}(\%)$	$\mathcal{F}(\%)$	$\mathcal{J}\&\mathcal{F}(\%)$	Time (s)
MEM	82.4	87.4	84.9	0.04
W/O MEM	80.2	82.5	81.3	0.04

4.1. Datasets

The *DAVIS 2016 & 2017* [27, 28] datasets are pixel-level VOS datasets. Their annotations are binary annotations that only distinguish between the front and back views, including camera shake, high-speed motion, and other complex application scenarios. DAVIS 2016 is a single target dataset with 30 training sets and 20 validation sets, including portraits, animals, and natural landscapes. DAVIS 2017 is a multi-objective dataset with 150 datasets.

The *YouTube-VOS 2018* dataset [42] is the largest video target segmentation dataset yet, containing 4453 video clips and 94 object classifications. This dataset can validate the model's segmentation performance for multiple types of targets.

4.2. Implementation Details

Our model is trained on a 12GB 2080Ti GPU using the Adam optimization function and PyTorch as the programming language.

Fig. 7(a) shows our training process, starting with pre-training of the model video segmentation on the image dataset, followed by the main training on the video dataset.

4.2.1 Pre-training on Image Datasets

Training with a static image database compensates for the lack of frames in the video database, and avoids over-fitting caused by a lack of training data. This method assumes no temporal relationship between images, and uses static picture datasets to train the video object segmentation models. Previous works used static images to train their networks, and we took a similar approach. The specific implementation applies random affine transformations [26] to various

Table 3. Comparison using DAVIS 2016 and DAVIS 2017 validation sets. Results for online (OL) and non-online methods are sorted by $\mathcal{J}\&\mathcal{F}$ mean. The three best scores are indicated in red, blue and yellow, respectively (same for other tables).

Method	OL	Frames/Second	DAVIS-2016			DAVIS-2017		
			$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$ Mean	$\mathcal{F}$ Mean
OSVOS [2]	✓	0.14	80.2	79.8	80.6	60.3	56.6	63.9
CINM [1]	✓	<0.03	84.2	83.4	85.0	70.6	67.2	74.0
OnAVOS [35]	✓	0.07	85.5	86.1	84.9	65.4	61.6	69.1
OSVOS-S [22]	✓	0.22	86.6	85.6	87.5	-	-	-
PReMVOS [21]	✓	<0.03	**86.8**	84.9	**88.6**	**77.8**	**73.9**	**81.8**
GEM [15]		-	64.6	69.6	59.6	-	-	-
OSMN [43]		7.69	73.5	74.0	72.9	54.8	52.5	57.1
PML [3]		3.57	77.4	75.5	79.3	-	-	-
VidMatch [9]		3.12	-	81.0	-	-	56.5	-
FEELVOS [34]		2	81.7	80.3	83.1	69.1	65.9	72.3
RGMP [24]		7.69	81.8	81.5	82.0	66.7	64.8	68.6
AGAME [10]		14.28	81.9	81.5	82.2	70.0	67.2	72.7
BMVOS [5]		**50**	82.2	82.9	81.4	72.7	70.7	74.7
RANet [39]		7.69	85.5	85.5	85.4	65.7	63.2	68.2
STM [25]		6.66	86.5	84.8	88.1	71.6	69.2	74.0
GC [14]		**30**	86.6	**87.6**	85.7	71.4	69.3	73.5
STCN [4]		20	**91.6**	**90.8**	**92.5**	**85.4**	**82.2**	**88.6**
OURS		**28**	**90.2**	**91.0**	**89.4**	**84.9**	**82.4**	**87.4**

Table 4. Comparison using the YouTube-VOS validation set.

Method	Overall	$\mathcal{J}$ seen	$\mathcal{J}$ unseen	$\mathcal{F}$ seen	$\mathcal{F}$ unseen
RVOS [33]	56.8	63.6	45.5	67.2	51.0
OSVOS [2]	58.8	59.8	54.2	60.5	60.7
VSBMM [40]	64.5	70.0	62.5	66.2	59.3
PReMVOS [21]	66.9	71.4	56.5	75.9	63.7
AGAME [10]	66.1	67.8	60.8	-	-
BoLTVOS [36]	71.1	71.6	64.3	-	-
AGSS [16]	71.3	71.3	65.5	76.2	73.1
GC [14]	73.2	72.6	68.9	75.6	75.7
BMVOS [5]	73.9	73.5	68.5	77.4	76.0
STM [25]	79.4	79.7	72.8	84.2	80.9
JOINT [23]	**81.5**	**85.9**	**78.7**	**86.5**	**83.1**
STCN [4]	**83.0**	**81.9**	**86.5**	**77.9**	**85.7**
OURS	**82.0**	**83.2**	**78.3**	**80.6**	**85.9**

images. A video sequence composed of three frames is generated and used to train our network, making our network more robust and easier to adapt to different segmentation targets. We pre-trained our model on the CoCo dataset [17].

4.2.2 Main Training on Video Datasets

We use DAVIS 2016 [27], DAVIS 2017 [28], and YouTube-VOS 2018 [42] datasets for the main training to allow the model to learn real-world video data. The video datasets make up for the incoherence between pictures in the image dataset, allowing the model to learn the timing information in the video. Meanwhile, the diversity of video objects also enhances the robustness of the model. Thanks to the low memory usage of the model, we randomly take ten frames from the same video and feed them into the model in chronological order for training. This allows our model to learn as much information about the spatial variation of objects in continuous time as possible. Some frames are randomly skipped in the extraction of video frames. The range of skipped frames should not be too large, so that the MEM can be trained to the greatest extent. We set the skip range from 0 to 10.

4.2.3 Other Training Details

We randomly clipped input frames to a size of 384×384. We used the Adam optimizer with a fixed learning rate of

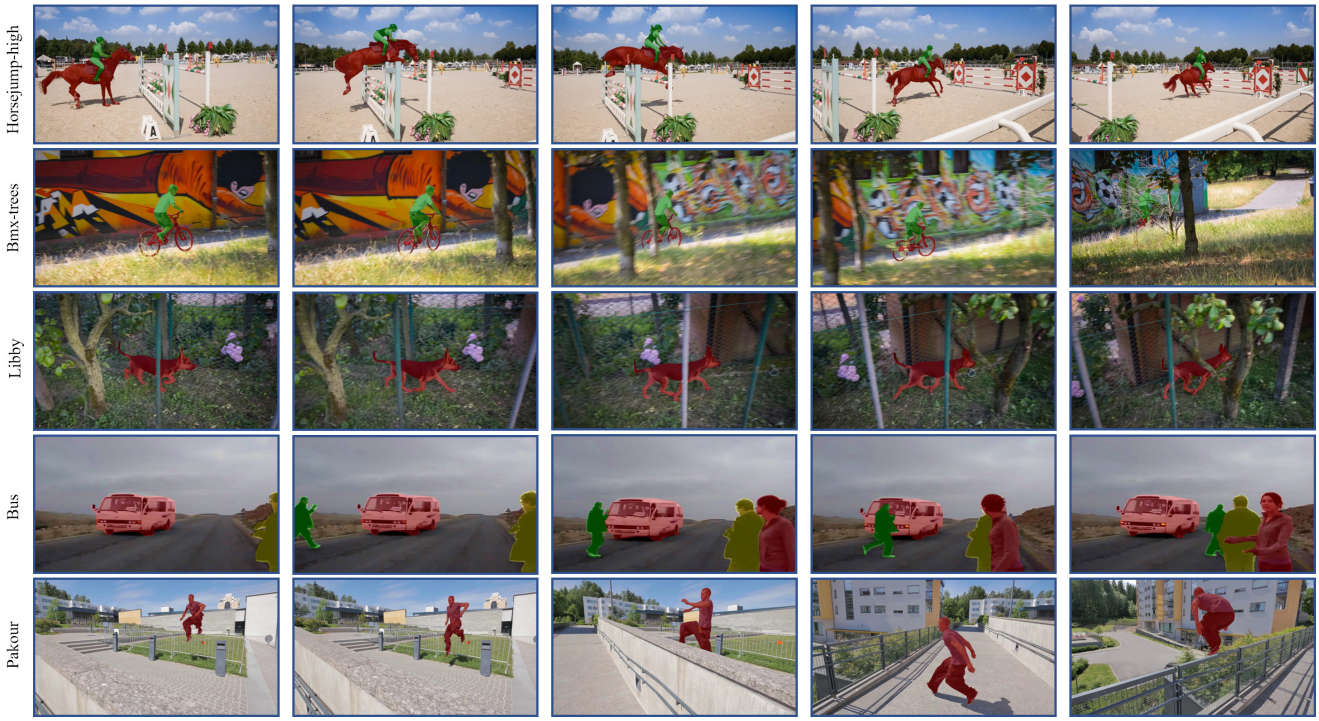


Figure 9. Further results from our method using the DAVIS 2017 validation set. It shows that on single object (rows 1, 3), the segmentation of the target is very accurate. On multi-target objects (rows 2, 4, 5), our method has good anti-interference ability and is not easily affected by the distractors.

10^{-5}. We froze the batch normalization layer during training. The mini-batch size was 4. Both pre-training and main-training used random affine transformations. The sampling intervals increased by 5 after every 20 epochs for DAVIS and YouTube-VOS.

4.3. Running Time Analysis

We compare the spatiotemporal memory methods with our method in terms of single-frame running time, using DAVIS 2017 as the test dataset, and the results appear in Fig. 1(b). Our method consumes 50% fewer resources in single-frame segmentation, and the running time is twice as fast compared to the spatiotemporal memory method. Thanks to our lightweight feature extraction network, the computational resources are reduced in the encoding phase. Furthermore, our global module alleviates the tight memory occupation in the matching phase.

4.4. Ablation Study

We performed ablation experiments using the DAVIS 2017 dataset to see how each module of our network contributes to the final results.

4.4.1 Pre-training and Main Training

Fig. 7(a) shows three training methods, the first one is to train with images only, the second one is to train with video datasets only, and the last one is the combination of pre-training and main training method we used. Fig. 7(b) shows the training accuracy curves for the three methods. The intriguing finding is that the accuracy of the model trained with only images is higher than that trained with only video sets, which illustrates the importance of the dataset size in the training process. These experiments show that the rich static image resources used in pre-training can help enhance our network's robustness, so we use both pre-training and main-training strategies for the model to achieve the best results.

4.4.2 Channel-wise Attention Module

Channel-wise attention reassigns the weights of all channels to enhance the feature representation of all foreground objects. We tested the accuracy performance with and without adding this module to the DAVIS 2017 dataset separately, and the results appear in Table 1. With the help of the channel attention module, The $\mathcal{J}$ mean and $\mathcal{F}$ mean are improved by 2.0% and 2.1%

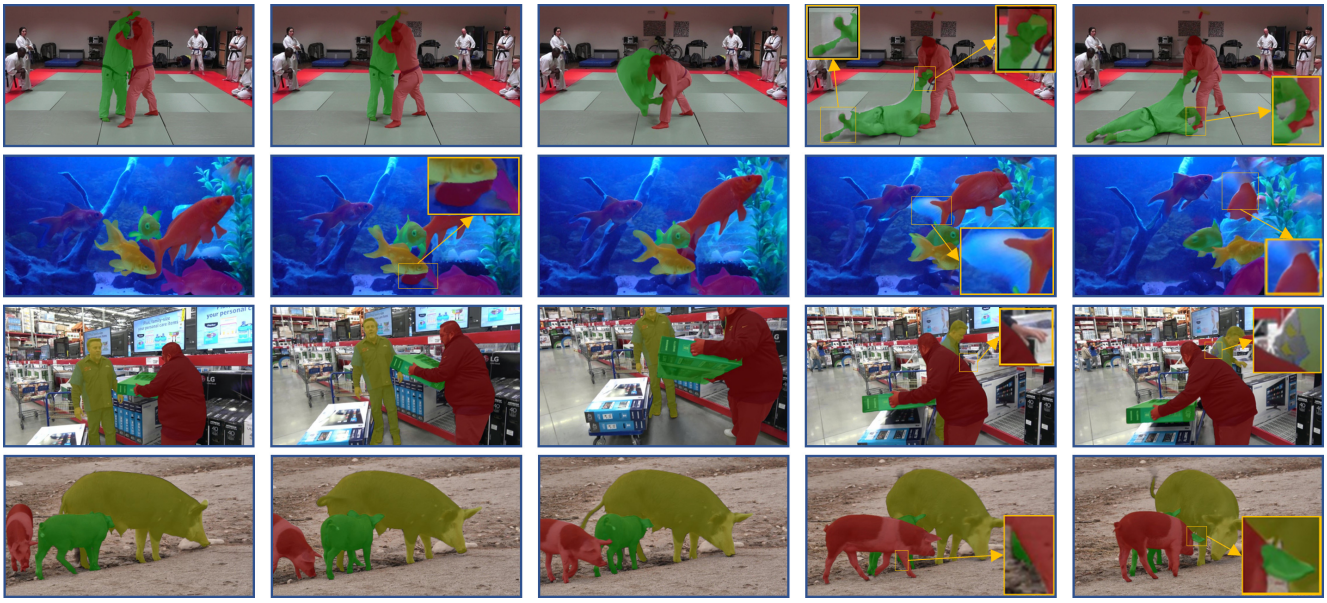


Figure 10. Imperfect segmentation results. Some imperfect segmentation results under extreme conditions. Generally our framework provides very good segmentation results even with occlusion, distractors, and complex object appearance.

4.4.3 Mask Embedding Module

The mask embedding module reduces the mismatching of target objects with a similar appearance. A comparison was performed with and without the module using the DAVIS 2017 dataset. It shows that the module can significantly prevent mismatching yet has little effect on computational efficiency, as shown in Table 2. In a multi-object video set, the target is more susceptible to interference from similar objects, and the improvement provided by the mask embedding becomes very obvious. When using the mask embedding module, $\mathcal{J}$ and $\mathcal{F}$ improve by 2.2% and 4.9%, respectively, while mask embedding does not affect speed. As Fig. 5 shows, the mask embedding module uses previous frames and their position encoding to help segment the current frame, which significantly reduces mismatching.

4.5. Comparisons to State-of-the-Art Methods

4.5.1 DAVIS 2016 (Single Object)

We compared our method with other SOTA techniques on the DAVIS 2016 benchmark, with single-object videos. including STM [25], STCN [4], RANet [39], and other recent methods. The results appear in Table 3, and ours has outstanding segmentation results while maintaining the real-time segmentation speed ($0.04s/frame$), with an average accuracy of 90.2%. Compared with STM [25], the single-frame segmentation speed is twice as fast, the average video segmentation speed is three times faster, and the $\mathcal{J}$ mean is 6.2% higher than STM [25].

The second column of Fig. 8 (break-dance) compares our segmentation performance with other methods on the DAVIS 2016 dataset. Due to the complexity of the scene, we can see that there are many distractors around the dancer. Both BMVOS [5] and RANet [39] treat some spectators as segmentation subjects, leading to mis-segmentation. Thanks to our mask embedding module, our model focuses on the target dancer and achieves a good segmentation effect. JOINT [23] and STM [25] methods are not affected by distractors in this scene. However, this is a case of imperfect segmentation accuracy, and these two methods do not segment the target completely.

Fig. 7(c) visualizes the comparison between our method and other methods in speed and accuracy. The horizontal coordinates represent FPS, and the vertical coordinates represent the accuracy under DAVIS 2016. It shows that we have achieved a satisfactory balance in speed and segmentation performance.

4.5.2 DAVIS 2017 (Multiple Object)

We compare with other SOTA methods on the DAVIS 2017 dataset, and the results appear in Table 3. Multi-object scenarios are more challenging than single-target scenarios, our segmentation performance is very close to STCN [4], and our method is 20% faster than STCN [4] in the dataset at $480p$ resolution on average. We achieved an average accuracy of 84.9%. the $\mathcal{J}$ mean and $\mathcal{J}$ mean are 13.2% and 13.4% higher than STM [25] respectively.

Fig. 8 compares our result plot with other methods. In the first column (Bike-packing), the surrounding environment of the target is more complex. Both BMVOS [5] and STM [25] have mismatches, and RANet [39] has in-

complete segmentation. Our method accurately segmented bike and people. The third column (Dogs-jump) shows the segmentation performance when similar objects exist, and there are two similar dogs in the scene. BMVOS [5] and RANet [39] are not good at identifying similar objects, our model gives convincing results thanks to the robustness of the model. Fig. 9 shows additional results from our method using the DAVIS 2017 validation set.

Fig. 7(c) shows that the horizontal coordinate represents frames per second. The vertical coordinate represents each model's accuracy under the single-object dataset. The bubble size represents the accuracy of each model under the multi-model dataset. The more significant the bubble indicates greater progress. While our method guarantees the speed and accuracy of single-object segmentation, the accuracy of multi-object segmentation is also satisfactory.

4.5.3 YouTube-VOS

One of the features of the YouTube-VOS dataset is that there are some unseen targets in the validation set. Table 4 compares several methods using this dataset. STM [25] again achieved high scores in this test. Our framework notably improves upon STM [25] and achieves high scores on seen and unseen object segmentation.

4.6. Limitations

However, our method still has room for improvement, and Fig. 10 shows some of our imperfect results. When the shape of the target object changes drastically (see row 1, columns 4, 5), the model cannot identify the boundaries of adjacent objects well. Since our method is based on visual information, the model cannot segment the boundaries of objects very well when the target object and the surrounding environment are very similar, such as the transparent fins of goldfish (see row 2, columns 2, 4, 5). As the mask embedding module uses the segmentation masks of the previous frames as the spatial constraints of the current segmentation, when the target object does not appear in the picture for a long time (see row 3, columns 4), the module cannot handle it well This situation. Thanks to the robustness of the model, this situation is generally corrected right away (see row 3, columns 5). When multiple similar objects appear in the same frame for a long time (see row 4, columns 4, 5), mis-segmentation may also occur.

To sum up, our model has some imperfect segmentation results under extreme conditions. Overall, our model can provides impressive segmentation results in the vast majority of cases, even with occlusion, distractors, and complex object appearance. Our network also achieves an excellent balance between accuracy and speed.

5. Conclusion

We design a real-time high-resolution video object segmentation model, which achieves fast feature extraction and matching by a modified global module. Moreover, the algorithm enhances the representational display of foreground objects by the channel attention module and finally eliminates the effect of confusing objects by the mask embedding module. Our model saves 50% computational resource consumption on single-frame segmentation compared to the temporal memory algorithm. It can achieve a segmentation speed of 25fps on $2K$ videos and has SOTA performance on standard datasets. We hope it will serve as a vital cornerstone for future efficient video target segmentation algorithms.

Acknowledgement

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 61802197, 62072449, and 61632003), the Science and Technology Development Fund, Macau SAR (Grant Nos. 0018/2019/AKP and SKL-IOTSC(UM)-2021-2023), the Guangdong Science and Technology Department (Grant No. 2020B1515130001), and University of Macau (Grant No. MYRG2020-00253-FST, MYRG2022-00059-FST).

References

[1] L. Bao, B. Wu, and W. Liu. Cnn in mrf: Video object segmentation via inference in a cnn-based higher-order spatio-temporal mrf. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5977–5986, 2018. 9

[2] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 221–230, 2017. 1, 3, 9

[3] Y. Chen, J. Pont-Tuset, A. Montes, and L. Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1189–1198, 2018. 1, 3, 9

[4] H. K. Cheng, Y.-W. Tai, and C.-K. Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 3, 4, 9, 11

[5] S. Cho, H. Lee, M. Kim, S. Jang, and S. Lee. Pixel-level bijective matching for video object segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 129–138, 2022. 3, 8, 9, 11, 12

[6] K. Fu, Y. Jiang, G.-P. Ji, T. Zhou, Q. Zhao, and D.-P. Fan. Light field salient object detection: A review and benchmark. *Computational Visual Media*, pages 1–26, 2022. 1

[7] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M.

Hu. Attention mechanisms in computer vision: A survey. *Computational Visual Media*, pages 1–38, 2022. 1

[8] L. Hu, P. Zhang, B. Zhang, P. Pan, Y. Xu, and R. Jin. Learning position and target consistency for memory-based video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4144–4154, 2021. 3

[9] Y.-T. Hu, J.-B. Huang, and A. G. Schwing. Videomatch: Matching based video object segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 54–70, 2018. 1, 3, 9

[10] J. Johnander, M. Danelljan, E. Brissman, F. S. Khan, and M. Felsberg. A generative appearance model for end-to-end video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8953–8962, 2019. 9

[11] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for video object segmentation. *International Journal of Computer Vision*, 127(9):1175–1197, 2019. 2, 3

[12] Y. Lan, Y. Duan, C. Liu, C. Zhu, Y. Xiong, H. Huang, and K. Xu. Arm3d: Attention-based relation module for indoor 3d object detection. *Computational Visual Media*, 8(3):395–414, 2022. 1

[13] X. Li, S. Liu, S. De Mello, X. Wang, J. Kautz, and M.-H. Yang. Joint-task self-supervised learning for temporal correspondence. *arXiv preprint arXiv:1909.11895*, 2019. 2, 3

[14] Y. Li, Z. Shen, and Y. Shan. Fast video object segmentation using the global context module. *arXiv preprint arXiv:2001.11243*, 2020. 2, 3, 4, 9

[15] Y. Li, L. Wen, M.-C. Chang, and S. Lyu. Graph-to-graph energy minimization for video object segmentation. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8, 2019. 9

[16] H. Lin, X. Qi, and J. Jia. Agss-vos: Attention guided single-shot video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3949–3957, 2019. 9

[17] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014. 9

[18] Y. Liu, K. Wang, H. Lan, and L. Lin. Temporal contrastive graph for self-supervised video representation learning. *arXiv preprint arXiv:2101.00820*, 2021. 3

[19] Y. Liu, K. Wang, G. Li, and L. Lin. Semantics-aware adaptive knowledge distillation for sensor-to-vision action recognition. *IEEE Transactions on Image Processing*, 30:5573–5588, 2021. 1

[20] Y. Liu, Y.-S. Wei, H. Yan, G.-B. Li, and L. Lin. Causal reasoning meets visual representation learning: A prospective study. *Machine Intelligence Research*, pages 1–27, 2022. 1

[21] J. Luiten, P. Voigtlaender, and B. Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. *arXiv preprint arXiv:1807.09190*, 2018. 9

[22] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. Video object segmentation without temporal information. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1515–1530, 2018. 1, 3, 9

[23] Y. Mao, N. Wang, W. Zhou, and H. Li. Joint inductive and transductive learning for video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9670–9679, 2021. 8, 9, 11

[24] S. W. Oh, J.-Y. Lee, K. Sunkavalli, and S. J. Kim. Fast video object segmentation by reference-guided mask propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7376–7385, 2018. 9

[25] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9226–9235, 2019. 2, 3, 4, 7, 8, 9, 11, 12

[26] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung. Learning video object segmentation from static images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2663–2672, 2017. 2, 3, 8

[27] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 8, 9

[28] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 8, 9

[29] H. Seong, J. Hyun, and E. Kim. Kernelized memory network for video object segmentation. *arXiv preprint arXiv:2007.08270*, 2020. 2, 3, 4

[30] Y. Song, F. Tang, W. Dong, and C. Xu. Non-dominated sorting based multi-page photo collage. *Computational Visual Media*, 8(2):199–212, 2022. 1

[31] L. Tang, K. Chen, C. Wu, Y. Hong, K. Jia, and Z.-X. Yang. Improving semantic analysis on point clouds via auxiliary supervision of local geometric priors. *IEEE Transactions on Cybernetics*, 2020. 3

[32] Y. Tang, Y. Zhang, X. Han, F.-L. Zhang, Y.-K. Lai, and R. Tong. 3d corrective nose reconstruction from a single image. *Computational Visual Media*, 8(2):225–237, 2022. 1

[33] C. Ventura, M. Bellver, A. Girbau, A. Salvador, F. Marques, and X. Giro-i Nieto. Rvos: End-to-end recurrent network for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5277–5286, 2019. 9

[34] P. Voigtlaender, Y. Chai, F. Schroff, H. Adam, B. Leibe, and L.-C. Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019. 9

[35] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. *arXiv preprint arXiv:1706.09364*, 2017. 1, 3, 9

[36] P. Voigtlaender, J. Luiten, and B. Leibe. Boltvos: Box-level tracking for video object segmentation. *arXiv preprint arXiv:1904.04552*, 2019. 9

[37] H. Wang, X. Jiang, H. Ren, Y. Hu, and S. Bai. Swiftnet: Real-time video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1296–1305, 2021. 2, 3, 4

[38] X. Wang, A. Jabri, and A. A. Efros. Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2566–2576, 2019. 2, 3

[39] Z. Wang, J. Xu, L. Liu, F. Zhu, and L. Shao. Ranet: Ranking attention network for fast video object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3978–3987, 2019. 8, 9, 11, 12

[40] S. Wehrwein and R. Szeliski. Video segmentation with background motion models. In *BMVC*, volume 245, page 246, 2017. 9

[41] H. Xie, H. Yao, S. Zhou, S. Zhang, and W. Sun. Efficient regional memory network for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1286–1295, 2021. 3, 4

[42] N. Xu, L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, and T. Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018. 8, 9

[43] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6499–6507, 2018. 9

[44] Z.-X. Yang, L. Tang, K. Zhang, and P. K. Wong. Multi-view cnn feature aggregation with elm auto-encoder for 3d shape recognition. *Cognitive Computation*, 10(6):908–921, 2018. 3

[45] S. Yu, J. Xiao, B. Zhang, E. G. Lim, and Y. Zhao. Fast pixel-matching for video object segmentation. *Signal Processing: Image Communication*, 98:116373, 2021. 3

[46] X. Zeng, Z. Wu, X. Peng, and Y. Qiao. Joint 3d facial shape reconstruction and texture completion from a single image. *Computational Visual Media*, 8(2):239–256, 2022. 1

[47] M.-L. Zhang and Z.-H. Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007. 3

[48] P. Zhang, L. Hu, B. Zhang, P. Pan, Z. Yang, Y. Ding, Y. Wei, Y. Yang, H. Seong, J. Hyun, et al. Spatial constrained memory network for semi-supervised video object segmentation. In *The 2020 DAVIS Challenge on Video Object Segmentation-CVPR Workshops*, pages 1–4, 2020. 3, 4